How to create a basic Bash script to monitor the uptime of a website

Monitoring the uptime of a website is essential to ensure that your site is always accessible to visitors. With a Bash script, you can automate uptime checks and receive immediate notifications in case of downtime. Here is a step-by-step guide on how to create a Bash script to monitor the uptime of a website.

We can create the following code:

```
#!/bin/bash

# URL of the website to monitor
website_url="https://yourwebsite.com"

# Email to receive notifications
notify_email="youremail@example.com"

# Time interval between checks (in seconds)
delay_check=60

# Function to check the website's uptime
check_website() {
  local status_code=$(curl -o /dev/null -s -w "%{http_code}\n" "$website_url")
  local timestamp=$(date "+%Y-%m-%d %H:%M:%S")

  if [ "$status_code" -ne 200 ]; then
    echo "$timestamp - $website_url down. Status code: $status_code" | mail -s "Down" "$notify_email"
  else
    echo "$timestamp - $website_url up. Status code:
$status_code"
  }
}

# Infinite loop to check the site at regular
intervals
while true; do
  check_website
  sleep $delay_check
done
```

This script will start monitoring the uptime of the specified website and send an email notification whenever the site is unreachable (i.e., when the HTTP status code is not 200).

To make the file executable, type the following command in the terminal:

```
chmod a+x check-website.sh
```

You can customize several aspects of the script:

1. **Website URL**: Modify the `website_url` variable with the address of the site you want to monitor.
2. **Email for notifications**: Change the `notify_email` variable with your email address.
3. **Time interval**: Adjust the `delay_check` variable to change the frequency of checks (in seconds).

You can also completely remove the infinite loop and manage the interval between checks using a cronjob. The script would then be modified as follows:

```
#!/bin/bash

# URL of the website to monitor
website_url="https://yourwebsite.com"

# Email to receive notifications
notify_email="youremail@example.com"

# Function to check the website's uptime
check_website() {
  local status_code=$(curl -o /dev/null -s -w "%
{http_code}\n" "$website_url")
  local timestamp=$(date "+%Y-%m-%d %H:%M:%S")

  if [ "$status_code" -ne 200 ]; then
    echo "$timestamp - $website_url down. Status
code: $status_code" | mail -s "Down" "$notify_email"
  else
    echo "$timestamp - $website_url up. Status code:
$status_code"
  }
}

check_website
```

Then, by typing `crontab -e`, you can insert the following line:

```
* * * * * /bin/bash /usr/local/bin/check-website.sh >
/dev/null 2>&1
```

Conclusion

Monitoring your website's uptime is crucial to ensure service availability to your users. With a simple Bash script, you can automate this process and receive immediate notifications in case of issues. This allows you to quickly address any downtime, improving the reliability of your website.